Class 04: Linear Regression and Regularization

Martin Slawski

Volgenau School of Engineering
Department of Statistics

February 15th, 2018

Recap: Empirical Risk Minimization (ERM)

Setup:

- sample $\mathcal{D}_n = \{(X_i, Y_i)\}_{i=1}^n$,
- a loss function L,
- a function class $\mathcal{F}$

Empirical risk minimization:

$$\min_{f \in \mathcal{F}} R_{\text{emp}}(f) = \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)).$$

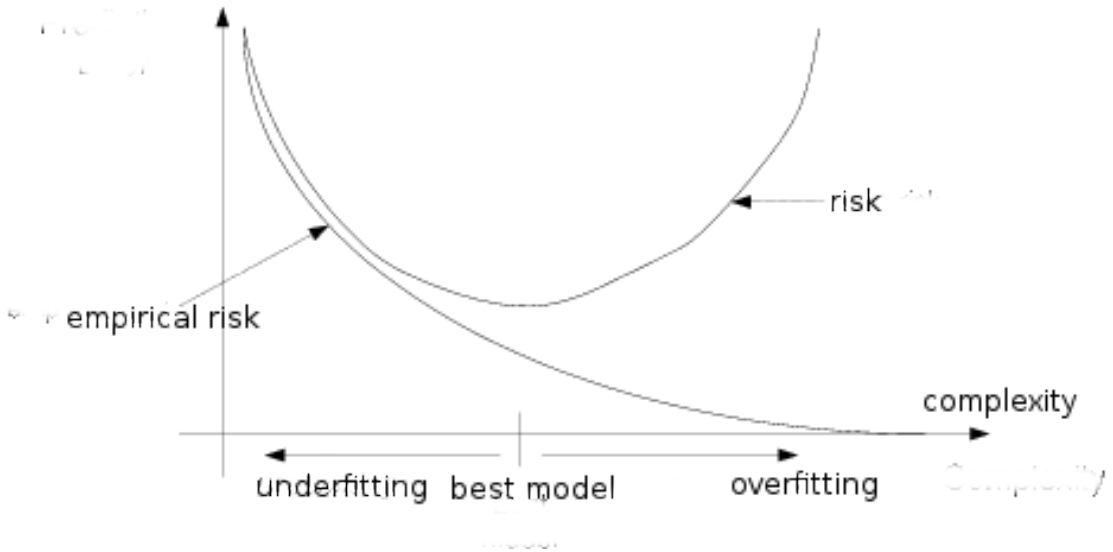
Let $\widehat{f}$ denote the empirical risk minimizer. We have seen that $R_{\text{emp}}(\widehat{f})$ can be a poor proxy for the generalization error

$$\mathbf{E}_{X,Y}[L(Y, \widehat{f}(X))|\widehat{f}].$$

Overfitting and Underfitting

Overfitting:
By choosing $\mathcal{F}$ "large enough", it easy to achieve a small empirical risk. However, the empirical risk minimizer typically does not generalize well to future data $(X, Y) \sim P$.

Underfitting:
On the other hand, if $\mathcal{F}$ is too small, we may be unable to capture important characteristics of the problem we want to learn.



Structural Risk Minimization

What is good a practical strategy for controlling the capacity of $\mathcal{F}$?

Structural Risk Minimization: Working with a nested sequence of increasingly complex function classes

$$\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \subset \mathcal{F}_N,$$

perform ERM separately for each class, and then pick the empirical risk minimizer from the least complex class with reasonable empirical error (i.e. within reach of more complex classes).

The underlying principle is known as **Occam's Razor** (William Ockham, 1287–1347) or **Principle of Parsimony**:

Among all models that explain the data well, take the simplest one.

Structural Risk Minimization

The idea of Structural Risk Minimization (SRM) is commonly used for model selection in regression models, often combined with criteria such as Akaike's information criterion (**AIC**) or Schwartz' Bayesian information criterion (**BIC**).

A problem with SRM is that it tends to be computationally inefficient:

- we have to compute the ERM for each function class under consideration
- the number of candidate function classes can be prohibitively large

Regularization

A conceptually closely related strategy is **Regularization**.

- we work with a large a function class $\mathcal{F}$,
- we add an additional term (the **regularizer**) to the empirical risk,
- the purpose of the regularizer is to penalize more complex hypotheses in $\mathcal{F}$.

This yields **regularized empirical risk minimization**:

$$\min_{f \in \mathcal{F}} R_{\text{emp}}(f) + \lambda \Omega(f),$$

where

- $\Omega : \mathcal{F} \to \mathbb{R}_+$ is called regularizer,
- $\lambda \geq 0$ is called **regularization parameter**.

Regularized empirical risk minimization

$$\min_{f \in \mathcal{F}} R_{\text{emp}}(f) + \lambda \Omega(f),$$

The regularization parameter controls the trade-off between

- achieving low empirical risk ($\lambda \to 0$),
- controlling model complexity ($\lambda \to \infty$).

From an optimization standpoint, λ can be interpreted as a Lagrangian multiplier for the constrained optimization problem

$$\min_{f \in \mathcal{F}} R_{\text{emp}}(f)$$
$$\text{subject to } \Omega(f) \leq r,$$

Regularized empirical risk minimization

Many popular machine learning algorithms can be put into the framework of regularized empirical risk minimization.

They differ by the choice of the loss L and the regularizer Ω.

Examples:

- Ridge Regression,
- Support Vector Classification,
- Support Vector Regression,
- Logistic Regression,
- Lasso, Elastic Net, Group Lasso

Regularization for Linear Regression

We want to learn a function that predicts Y from X.

More specifically, our target is

$$f_L^*(x) = \mathbf{E}[Y|X = x],\ x \in \mathcal{X},$$

i.e. the corresponding loss function is squared loss

$$L(Y, f(X)) = (Y - f(X))^2.$$

The function class $\mathcal{F}$ is given by

$$\mathcal{F} = \left\{ f :\ x \mapsto f(x) = \sum_{j=1}^{D} w_j \phi_j(x),\ \ w \in \mathbb{R}^D \right\},$$

where the $\{\phi_j\}_{j=1}^D$ are fixed functions $\mathcal{X} \to \mathbb{R}$.

Regularization for Linear Regression

The function class $\mathcal{F}$ is given by

$$\mathcal{F} = \left\{ f : \; x \mapsto f(x) = \sum_{j=1}^{D} w_j \phi_j(x), \;\; w \in \mathbb{R}^D \right\},$$

where the $\{\phi_j\}_{j=1}^D$ are fixed functions $\mathcal{X} \to \mathbb{R}$.

The use of a input transformation (or **feature map**)

$$\begin{aligned} \mathcal{X} &\to \mathbb{R}^D \\ x &\mapsto \Phi(x) := \begin{pmatrix} \phi_1(x) \\ \vdots \\ \phi_D(x) \end{pmatrix} \end{aligned}$$

allows for additional modeling flexibility.

Regularization for Linear Regression

Examples of Input transformations:

(1) No transformation: keep the original features, and only these

$$\phi_1(x) = x_1, \ldots, \phi_D(x) = x_d, \quad D = d.$$

(2) Interactions up to order two

$$\begin{aligned}
&\phi_1(x) = x_1, \ldots, \phi_d(x) = x_d, \\
&\phi_{d+1}(x) = x_1 x_2, \ldots, \phi_D = x_{d-1} x_d, \quad D = d(d+1)/2.
\end{aligned}$$

(3) Polynomial terms ($\mathcal{X} \subseteq \mathbb{R}$)

$$\phi_1(x) = 1, \ \phi_2(x) = x, \ldots, \ \phi_D(x) = x^{D-1}, \phi_D(x) = x^D.$$

Regularization for Linear Regression

(4) Trigonometric polynomials ($\mathcal{X} \subseteq [0,1]$)

$$\phi_1(x) = 1,\ \phi_2(x) = \sin(2\pi x), \phi_3(x) = \cos(2\pi x), \ldots,$$
$$\phi_{D-1}(x) = \sin(2\pi N x), \phi_D(x) = \cos(2\pi N x).$$

for some positive integer N.

many more possibilities: splines, wavelets, fractional polynomials, ...

Regularization for Linear Regression

Note: linear regression does not exclude nonlinear transformations of the inputs.

Linear Regression because

$$f(x) = \sum_{j=1}^{D} \phi_j(x) w_j = \Phi(x)^\top w, \quad w = \begin{pmatrix} w_1 \\ \vdots \\ w_D \end{pmatrix}$$

is a linear function in w.

Least Squares Regression

With $L(Y, f(X)) = \{Y - f(X)\}^2$, empirical risk minimization amounts to least squares estimation.

Denote by

$$\mathbf{\Phi} = \begin{pmatrix} \Phi(X_1)^\top \\ \vdots \\ \Phi(X_n)^\top \end{pmatrix} = \begin{pmatrix} \phi_1(X_1) & \dots & \phi_D(X_1) \\ \vdots & \dots & \vdots \\ \phi_1(X_n) & \dots & \phi_D(X_n). \end{pmatrix}$$

the $\mathbb{R}^{n \times D}$ **feature matrix** or **design matrix**, and let

$$\mathbf{y} = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix}$$

be the vector of outputs.

Least Squares Regression

Unless stated explicitly, there is no intercept in the model – i.e. the matrix $\mathbf{\Phi}$ does not contain a column of ones.

Note that we do not need an intercept whenever $\mathbf{y}$ and all columns of $\mathbf{\Phi}$ are centered (i.e., they sum up to zero).

Least Squares Regression

$$\min_{f \in \mathcal{F}} R_{\mathsf{emp}}(f) = \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i))$$

$$= \min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (Y_i - \Phi(X_i)^\top w)^2$$

$$= \min_{w \in \mathbb{R}^D} \frac{1}{n} \|\mathbf{y} - \mathbf{\Phi} w\|_2^2$$

As in Homework 0, it can be shown any $\widehat{w}_{\mathsf{LS}}$ satisfying the *normal equations*

$$\frac{1}{n} \mathbf{\Phi}^\top \mathbf{\Phi} \widehat{w}_{\mathsf{LS}} = \frac{1}{n} \mathbf{\Phi}^\top \mathbf{y}$$

is optimal. $\widehat{w}_{\mathsf{LS}}$ is called a least squares estimator. It is unique whenever $\mathbf{\Phi}$ is non-singular $\Leftrightarrow \mathbf{\Phi}^\top \mathbf{\Phi}$ is invertible. In this case,

$$\widehat{w}_{\mathsf{LS}} = (\mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{y}.$$

Least Squares Regression

Is least squares estimation a good idea?

Least squares is prone to overfit whenever D becomes large relative to n.

In the scenario $D \geq n$, it is possible that $\texttt{range}(\mathbf{\Phi}) = \mathbb{R}^n$, where $\texttt{range}(\mathbf{\Phi})$ denotes the column space of $\mathbf{\Phi}$. In this situation,

$$\mathbf{\Phi}\widehat{w}_{\mathsf{LS}} = \mathbf{y},$$

i.e., we perfectly fit the observed sample, but likely do poorly on unseen data.

Least Squares Regression

Example: Pima Indian Diabetes data set.

$n = 442$ diabetes patients

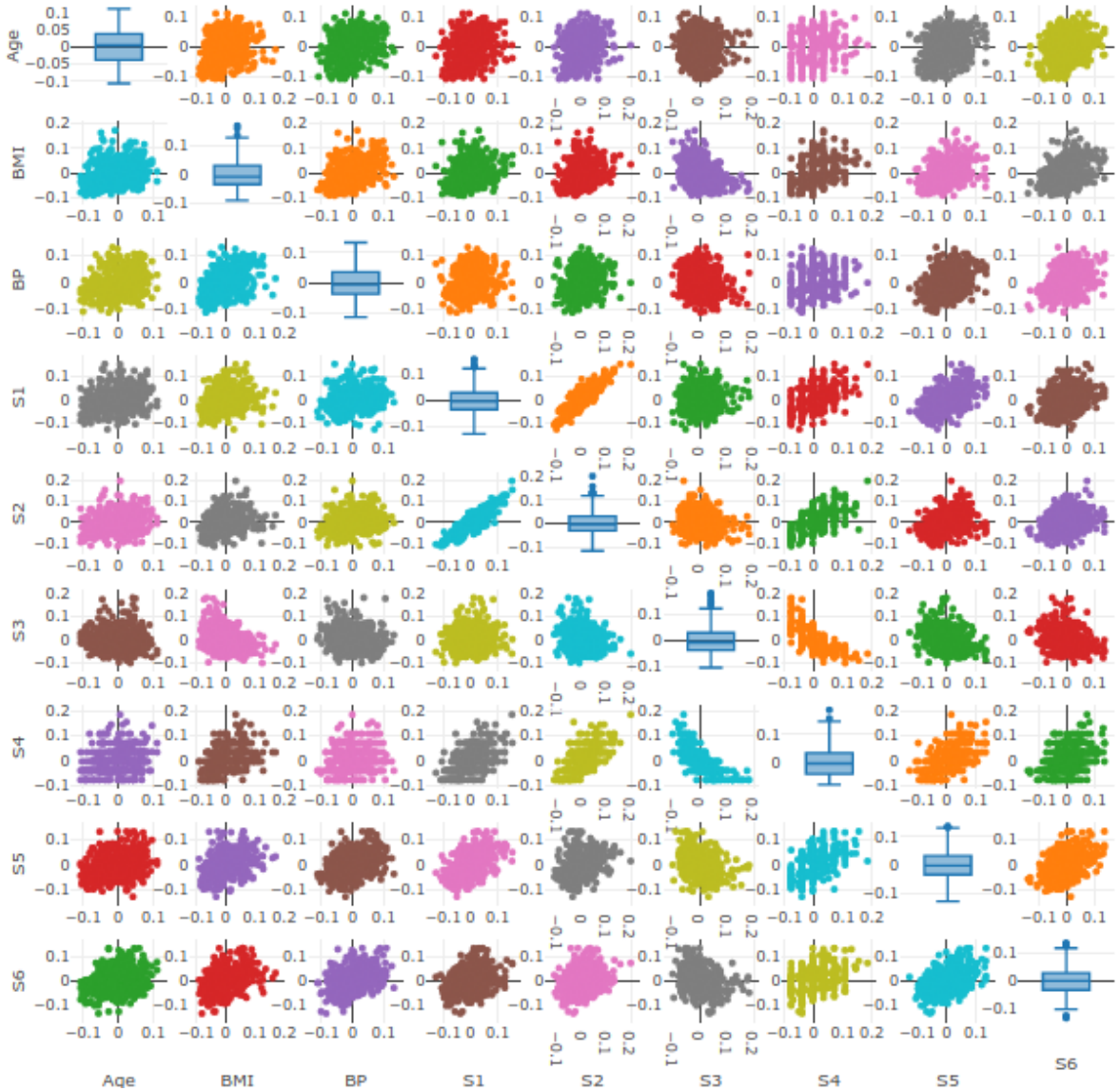
Target variable to be predicted is a measure of disease progression.

Covariates ($d = 10$):

- age
- sex
- body mass index
- average blood pressure
- six blood serum measurements

Least Squares Regression

Scatterplot matrix of continuous variables (all except for sex):



Least Squares Regression

Correlation matrix:

$$\begin{bmatrix}
\circ & 0.17 & 0.19 & 0.34 & 0.26 & 0.22 & -0.08 & 0.2 & 0.27 & 0.3 \\
\circ & \circ & 0.09 & 0.24 & 0.04 & 0.14 & -0.38 & 0.33 & 0.15 & 0.21 \\
\circ & \circ & \circ & 0.4 & 0.25 & 0.26 & -0.37 & 0.41 & 0.45 & 0.39 \\
\circ & \circ & \circ & \circ & 0.24 & 0.19 & -0.18 & 0.26 & 0.39 & 0.39 \\
\circ & \circ & \circ & \circ & \circ & 0.9 & 0.05 & 0.54 & 0.52 & 0.33 \\
\circ & \circ & \circ & \circ & \circ & \circ & -0.2 & 0.66 & 0.32 & 0.29 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & -0.74 & -0.4 & -0.27 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 0.62 & 0.42 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & 0.46 \\
\circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ
\end{bmatrix}$$

Least Squares Regression

We fit a linear regression model using all $d = 10$ covariates.

Training error	Leave-One-Out Error
2,860	2,988

To verify that least squares tends to overfit we randomly permute the rows of $\mathbf{\Phi}$ and append them as additional columns. This is repeated multiple times:

#permutations	D	Training error	Leave-One-Out Error
1	20	2,816	3,097
2	30	2,739	3,146
3	40	2,618	3,172
4	50	2,561	3,257

Least Squares Regression

This could be repeated by appending as many permuted versions of $\mathbf{\Phi}$ until the training error drops to zero.

The idea behind the permutations is that they break the correlation to the response $\mathbf{y}$; hence the extra columns take the role of "noise variables".

Least Squares Regression

In the following, we present a simplified analysis of least squares, that still conveys the main insights.

(I) Only the Y's are random, i.e

$$\mathcal{D}_n = \{(x_1, Y_1), \ldots, (x_n, Y_n)\},$$

where the $\{x_i\}_{i=1}^n$ are considered fixed.

Least Squares Regression

(II) Additive and homoskedastic noise

$$Y_i = f^*(x_i) + \varepsilon_i, \quad i = 1, \ldots, n,$$

where for the noise terms we assume that

- $\mathbf{E}[\varepsilon_i] = 0$,
- $\mathbf{E}[\varepsilon_i^2] = \sigma^2$, $i = 1, \ldots, n$,
- The $\{\varepsilon_i\}_{i=1}^n$ are uncorrelated.

Least Squares Regression

Suppose hypothetically we are also given a second data set, say, for the purpose of validation:

$$\mathcal{D}'_n = \{(x_1, Y'_1), \ldots, (x_n, Y'_n)\},$$

where

$$Y'_i = f^*(x_i) + \varepsilon'_i,$$

and ε'_i has the same distribution as ε_i, $i = 1, \ldots, n$.

Least Squares Regression

$\mathcal{D}_n$:

features	"ground truth"	noise	response
x_1	$f^*(x_1)$	ε_1	Y_1
$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_n	$f^*(x_n)$	ε_n	Y_n

Least Squares Regression

$\mathcal{D}'_n$:

features	"ground truth"	noise	response
x_1	$f^*(x_1)$	ε'_1	Y'_1
$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_n	$f^*(x_n)$	ε'_n	Y'_n

Least Squares Estimation

The statistical performance of $w \in \mathbb{R}^D$ can be quantified by

$$\mathsf{MSE}(w) = \mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i' - \Phi(x_i)^\top w)^2\right]$$

where $\mathbf{E}[\cdot]$ denotes the expectation with respect to ε'.

Least Squares Regression

Clearly, the optimal MSE is given by

$$\mathsf{MSE}(w^*) = \min_{w \in \mathbb{R}^D} \mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i' - \Phi(x_i)^\top w)^2\right]$$

This our benchmark, and we will compare how several estimators perform relative to this benchmark.

Least Squares Regression

It is not hard to show that

$$\mathsf{MSE}(w^*) = \min_{w \in \mathbb{R}^D} \mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i' - \Phi(x_i)^\top w)^2\right]$$

$$= \underbrace{\sigma^2}_{\text{noise}} + \underbrace{\min_{w \in \mathbb{R}^D} \frac{1}{n}\|\boldsymbol{f}^* - \boldsymbol{\Phi} w\|_2^2}_{\text{approximation error}},$$

where

$$\boldsymbol{f}^* = (f^*(x_1) \quad \dots \quad f^*(x_n))^\top.$$

Least Squares Estimation

One can show the following for the least squares estimator:

$$\mathbf{E}[\mathsf{MSE}(\widehat{w}_{\mathsf{LS}})] = \underbrace{\sigma^2}_{\text{noise}} + \underbrace{\min_{w \in \mathbb{R}^D} \frac{1}{n} \|\boldsymbol{f}^* - \boldsymbol{\Phi} w\|_2^2}_{\text{approximation error}} + \underbrace{\frac{\sigma^2 \mathsf{rank}(\boldsymbol{\Phi})}{n}}_{\text{estimation error}}$$

$$= \mathsf{MSE}(w^*) + \frac{\sigma^2 \mathsf{rank}(\boldsymbol{\Phi})}{n}$$

Note that since $\widehat{w}_{\mathsf{LS}}$ is random, so is $\mathsf{MSE}(\widehat{w}_{\mathsf{LS}})$ – that's why we take the expectation $\mathbf{E}[\cdot]$ w.r.t. ε from the data set $\mathcal{D}_n$ from which $\widehat{w}_{\mathsf{LS}}$ was obtained.

Least Squares Estimation

$$\mathbf{E}[\mathsf{MSE}(\widehat{w}_{\mathsf{LS}})] = \mathsf{MSE}(w^*) + \frac{\sigma^2 \mathsf{rank}(\mathbf{\Phi})}{n}$$

Note that typically

$$\mathsf{rank}(\mathbf{\Phi}) = \min\{D, n\}.$$

In this case, the result implies that least squares is not even statistically consistent unless D is small relative to n.

Least squares can be a poor method even if $n > D$ (and is useless if $D > n$).

Interlude: Singular Value Decomposition

A variety of properties of least squares regression and several other methods can be better understood in terms of the **singular value decomposition (SVD)** of $\mathbf{\Phi}$.

Without any additional assumption on $\mathbf{\Phi}$, we have

$$\mathbf{\Phi} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$$

- $\mathbf{U} \in \mathbb{R}^{n \times n}$ such that $\mathbf{U}^\top\mathbf{U} = \mathbf{U}\mathbf{U}^\top = I_n$,
- $\mathbf{S} \in \mathbb{R}^{n \times D}$ and $m = \min\{n, D\}$ such that

$$\mathbf{S} = \begin{bmatrix} \text{diag}(s_1, \ldots, s_m) & \mathbf{0}_{m \times (D-m)} \\ \mathbf{0}_{(n-m) \times m} & \mathbf{0}_{(n-m) \times (D-m)} \end{bmatrix},$$

- $\mathbf{V} \in \mathbb{R}^{D \times D}$ such that $\mathbf{V}^\top\mathbf{V} = \mathbf{V}\mathbf{V}^\top = I_D$.

Interlude: Singular Value Decomposition

$$\mathbf{\Phi} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$$

- $\mathbf{U} \in \mathbb{R}^{n \times n}$ such that $\mathbf{U}^\top \mathbf{U} = I_n$,
- $\mathbf{S} \in \mathbb{R}^{n \times D}$ and $m = \min\{n, D\}$ such that

$$\mathbf{S} = \begin{bmatrix} \text{diag}(s_1, \ldots, s_m) & \mathbf{0}_{m \times (D-m)} \\ \mathbf{0}_{(n-m) \times m} & \mathbf{0}_{(n-m) \times (D-m)} \end{bmatrix},$$

- $\mathbf{V} \in \mathbb{R}^{D \times D}$ such that $\mathbf{V}^\top \mathbf{V} = I_D$.

Without loss of generality, we assume that the **singular values**

$$s_1 \geq \ldots \geq s_m \geq 0$$

are ordered.

Interlude: Singular Value Decomposition

"Economy" SVD (default in `R`, optional in `python` and `MATLAB`) (recall $m = \min\{n, D\}$):

- $\mathbf{U} \in \mathbb{R}^{n \times m}$ such that $\mathbf{U}^\top \mathbf{U} = I_m$,
- $\mathbf{S} = \text{diag}(s_1, \ldots, s_m)$,
- $\mathbf{V} \in \mathbb{R}^{D \times m}$ such that $\mathbf{V}^\top \mathbf{V} = I_m$.

If $n \leq D$: $\mathbf{U}\mathbf{U}^\top = I_n$,

If $n \geq D$: $\mathbf{V}\mathbf{V}^\top = I_D$.

Interlude: Singular Value Decomposition

Basic properties:

- The columns of $\mathbf{U}$ are called **left singular values**
- The number of non-zero singular values equals $\text{rank}(\mathbf{\Phi}) \leq m$
- The columns of $\mathbf{V}$ are called **right singular values**

Moreover, we have

$$\mathbf{\Phi}^\top \mathbf{\Phi} = \mathbf{V}\mathbf{S}^2\mathbf{V}^\top = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top, \quad \mathbf{\Lambda} = \text{diag}(\underbrace{s_1^2}_{\lambda_1}, \ldots, \underbrace{s_m^2}_{\lambda_m})$$

i.e., the columns of $\mathbf{V}$ are the eigenvectors and $\lambda_1 \geq \ldots \geq \lambda_m \geq 0$ are the eigenvalues of $\mathbf{\Phi}^\top \mathbf{\Phi}$.

Interlude: Singular Value Decomposition

Finally, it can be shown that

$$\mathbf{\Phi}\widehat{w}_{\mathsf{LS}} = \mathbf{U}\mathbf{U}^\top \mathbf{y},$$

i.e., $\mathbf{U}\mathbf{U}^\top$ equals the "Hat matrix" of the least squares fit.

Interlude: Singular Value Decomposition

Let $\|M\|_F = \sqrt{\sum_{i,j} M_{ij}^2}$ denote the Frobenius norm of a matrix M.

Eckart-Young Theorem (1936)

For $r \leq m$, consider the minimization problem

$$\min_{\mathbf{Z}\text{: rank}(\mathbf{Z}) \leq r} \|\mathbf{\Phi} - \mathbf{Z}\|_F^2 .$$

Then an optimal solution for $\mathbf{Z}$ is given by

$$\mathbf{\Phi}_r = \mathbf{U}_r \mathbf{S}_r \mathbf{V}_r^\top ,$$

where $\mathbf{U}_r$ and $\mathbf{V}_r^\top$ contain the r columns of $\mathbf{U}$ respectively rows of $\mathbf{V}^\top$ corresponding to the largest r singular values, and

$$\mathbf{S}_r = \text{diag}(s_1, \ldots, s_r).$$

Interlude: Singular Value Decomposition

The matrix $\mathbf{\Phi}_r = \mathbf{U}_r \mathbf{S}_r \mathbf{V}_r^\top$ is called the best rank-r approximation to $\mathbf{\Phi}$.

It is not hard to show that

$$\|\mathbf{\Phi} - \mathbf{\Phi}_r\|_F^2 = \sum_{j:\, j>r} s_j^2,$$

and thus

$$\texttt{range}(\mathbf{\Phi}_r) \approx \texttt{range}(\mathbf{\Phi})$$

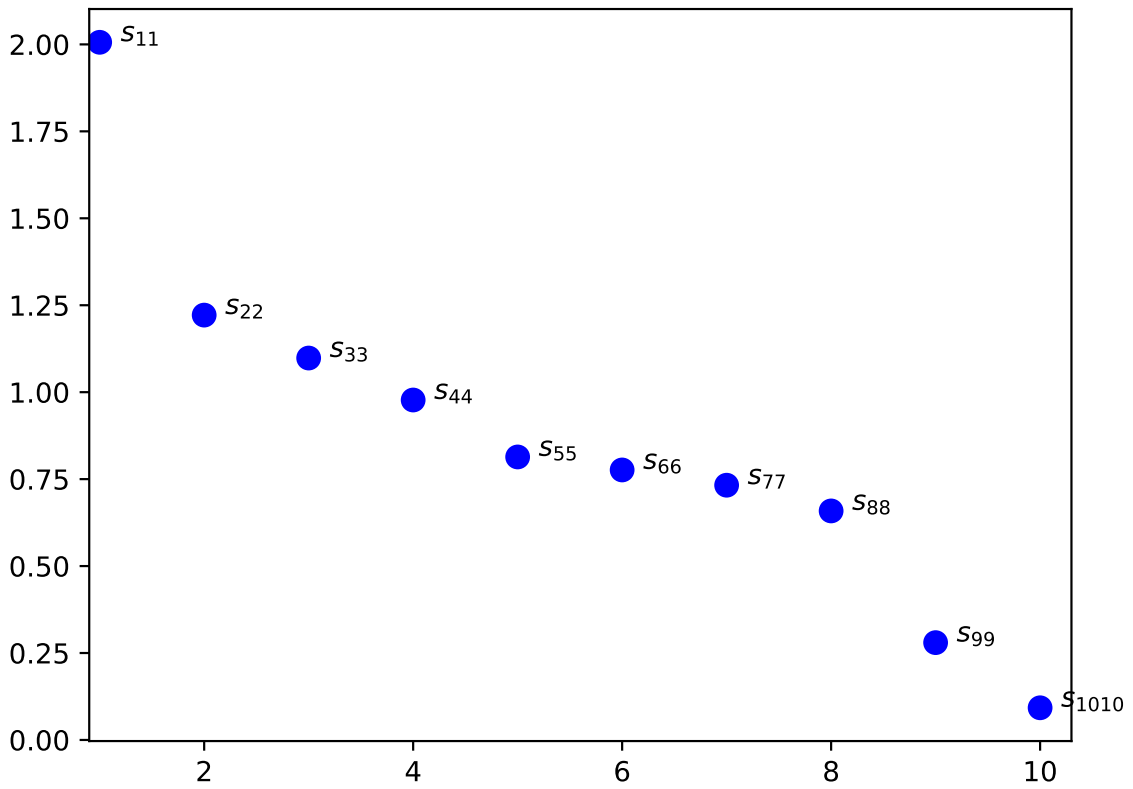
if the term

$$\sum_{j:\, j>r} s_j^2$$

is small.

Interlude: Singular Value Decomposition

Plot of the sequence of singular values for the diabetes data set:



Such plot is often called "Screeplot".

Interlude: Singular Value Decomposition

One way of improving over standard least squares is to work with $\mathbf{\Phi}_r$ respectively $\mathbf{U}_r$ (we have $\texttt{range}(\mathbf{\Phi}_r) = \texttt{range}(\mathbf{U}_r)$) instead of $\mathbf{\Phi}$.

Denote by $\widehat{w}_r$ the least squares estimator corresponding to $\mathbf{\Phi}_r$.

One can show that

$$\mathbf{E}[\mathsf{MSE}(\widehat{w}_r)] = \mathsf{MSE}(w^*) + \frac{1}{n}\sum_{j>r} s_j^2\{\alpha_j^*\}^2 + \frac{\sigma^2 r}{n}, \quad \alpha^* := V^\top w^*.$$

Hence if $r \ll D$ and $\sum_{j>r} s_j^2$ is small, the MSE gets substantially reduced.

Ridge Regression

Suppose $D > n$. Then the least squares estimator defined by the normal equations

$$\frac{1}{n}\mathbf{\Phi}^\top \mathbf{\Phi} \widehat{w}_{\mathsf{LS}} = \frac{1}{n}\mathbf{\Phi}^\top \mathbf{y}$$

is not unique.

One way of enforcing a unique solution is by adding multiple of the identity λI_D, $\lambda > 0$, to $\mathbf{\Phi}^\top \mathbf{\Phi}$: this yields

$$\left(\frac{1}{n}\mathbf{\Phi}^\top \mathbf{\Phi} + \lambda I_D\right) \widehat{w}_{\mathsf{ridge}} = \frac{1}{n}\mathbf{\Phi}^\top \mathbf{y}$$

$$\Rightarrow \ \widehat{w}_{\mathsf{ridge}} = \left(\frac{1}{n}\mathbf{\Phi}^\top \mathbf{\Phi} + \lambda I_D\right)^{-1} \frac{1}{n}\mathbf{\Phi}^\top \mathbf{y}$$

Ridge Regression

Equivalently, we can define the ridge estimator by

$$\widehat{w}_{\text{ridge}} = \underset{w \in \mathbb{R}^D}{\operatorname{argmin}} \frac{1}{n} \|\mathbf{y} - \mathbf{\Phi} w\|_2^2 + \lambda \|w\|_2^2$$

which is equivalent to finding

$$\underset{f \in \mathcal{F}}{\operatorname{argmin}} R_{\text{emp}}(f) + \lambda \Omega(f),$$

where

- R_{emp} is w.r.t. squared loss,
- $\mathcal{F}$ is the linear class on slide 9,
- $\Omega(f) = \Omega\left(\sum_{j=1}^D w_j \phi_j\right) = \|w(f)\|_2^2$

Ridge Regression

It can be shown that the ridge regression fit can be expressed as

$$\mathbf{\Phi}\widehat{w}_{\text{ridge}} = \mathbf{U}\ \text{diag}\left(\frac{\lambda_1}{\lambda_1+\lambda}, \ldots, \frac{\lambda_m}{\lambda_m+\lambda}\right)\ \mathbf{U}^\top \mathbf{y},$$

recalling that $\lambda_j = s_j^2$, $j = 1, \ldots, m$.

This is to be compared to

$$\mathbf{\Phi}\widehat{w}_{\text{LS}} = \mathbf{U}\mathbf{U}^\top \mathbf{y},$$

Ridge Regression

We can think of each of the columns of $\mathbf{U}$ as one specific direction among m directions associated with $\texttt{range}(\mathbf{\Phi})$.

Loosely speaking, the corresponding eigenvalues $\{\lambda_j\}_{j=1}^m$ represent the "prominence" of these directions in $\texttt{range}(\mathbf{\Phi})$.

The larger λ_j, the more "prominent" the corresponding direction.

Ridge Regression

$$\mathbf{\Phi}\widehat{w}_{\text{ridge}} = \mathbf{U}\ \text{diag}\left(\frac{\lambda_1}{\lambda_1+\lambda},\ldots,\frac{\lambda_m}{\lambda_m+\lambda}\right)\ \mathbf{U}^\top\mathbf{y},$$

We may intepret

$$\frac{\lambda_j}{\lambda_j+\lambda}$$

as a weight assigned to direction j, $j=1,\ldots,m$.

The presence of λ in the denominator dampens the influence of directions with small λ_j on the regression fit much more strongly.

Ridge Regression

It can be shown that

$$\mathbf{E}[\mathsf{MSE}(\widehat{w}_{\mathsf{ridge}})] = \mathsf{MSE}(w^*) + \sum_{j=1}^{D} \{\alpha_j^*\}^2 \lambda_j \left(\frac{\lambda}{\lambda + \lambda_j}\right)^2, \quad \alpha^* := V^\top w^*,$$

$$+ \frac{\sigma^2}{n} \sum_{j=1}^{D} \left(\frac{\lambda_j}{\lambda_j + \lambda}\right)^2,$$

-

Ridge Regression works well if the first few eigenvalues, say $\lambda_1, \ldots, \lambda_r$, are dominant, while the rest $\lambda_{r+1}, \ldots, \lambda_D$ are small, and λ is chosen such that

$$\lambda_{r+1} \ll \lambda \ll \lambda_r.$$

Ridge Regression

$$\mathbf{E}[\mathsf{MSE}(\widehat{w}_{\mathsf{ridge}})] = \mathsf{MSE}(w^*) + \sum_{j=1}^{D} \{\alpha_j^*\}^2 \lambda_j \left(\frac{\lambda}{\lambda + \lambda_j}\right)^2, \quad \alpha^* := V^\top w^*,$$

$$+ \frac{\sigma^2}{n} \sum_{j=1}^{D} \left(\frac{\lambda_j}{\lambda_j + \lambda}\right)^2$$

-

Ridge Regression does not work well if the magnitude of the α_j^*'s is not well aligned with the size of the λ_j (i.e. large α_j^* for small λ_j).

-

In particular, ridge regression tends to be suboptimal for **sparse** w^*. Ridge regression does not promote a parsimonious model.

Ridge Regression

Example: Pima Indian Diabetes data set (**continued**).

$n = 442$ diabetes patients

Target variable to be predicted is a measure of disease progression.

Covariates ($d = 10$):

- age
- sex
- body mass index
- average blood pressure
- six blood serum measurements

Ridge Regression

This time, we fit a "quadratic model", i.e. in addition to the original 10 covariates, we consider quadratic terms (except for sex)

$$X_1^2, X_3^2, \ldots, X_d^2,$$

and all first-order interactions

$$X_1X_2, X_1X_3, \ldots, X_{d-1}X_d.$$

This yields $D = 10 + 9 + \binom{10}{2} = 64$.

$D = 64$ is fairly large number of features relative to only $n = 442$ samples. Least squares cannot be expected to work well.

Ridge Regression

Important "pre-processing" steps before applying ridge regression:

1. Center $\mathbf{y}$ and the columns of $\mathbf{\Phi}$:

$$y_i \leftarrow y_i - \frac{1}{n}\sum_{i'=1}^{n} y_{i'}, \qquad \mathbf{\Phi}_{ij} \leftarrow \mathbf{\Phi}_{ij} - \frac{1}{n}\sum_{i'=1}^{n} \mathbf{\Phi}_{i'j},$$

$i = 1, \ldots, n$, $j = 1, \ldots, D$.

2. Scale the columns of $\mathbf{\Phi}$ so that their norms are identical:

$$\mathbf{\Phi}_{\bullet j} \leftarrow \mathbf{\Phi}_{\bullet j} / \|\mathbf{\Phi}_{\bullet j}\|_2, \; j = 1, \ldots, D.$$

Ridge Regression

On the importance of column normalization when using ridge regression:

If the features are not on the same scale, we essentially penalize each feature differently. This is typically not appropriate.

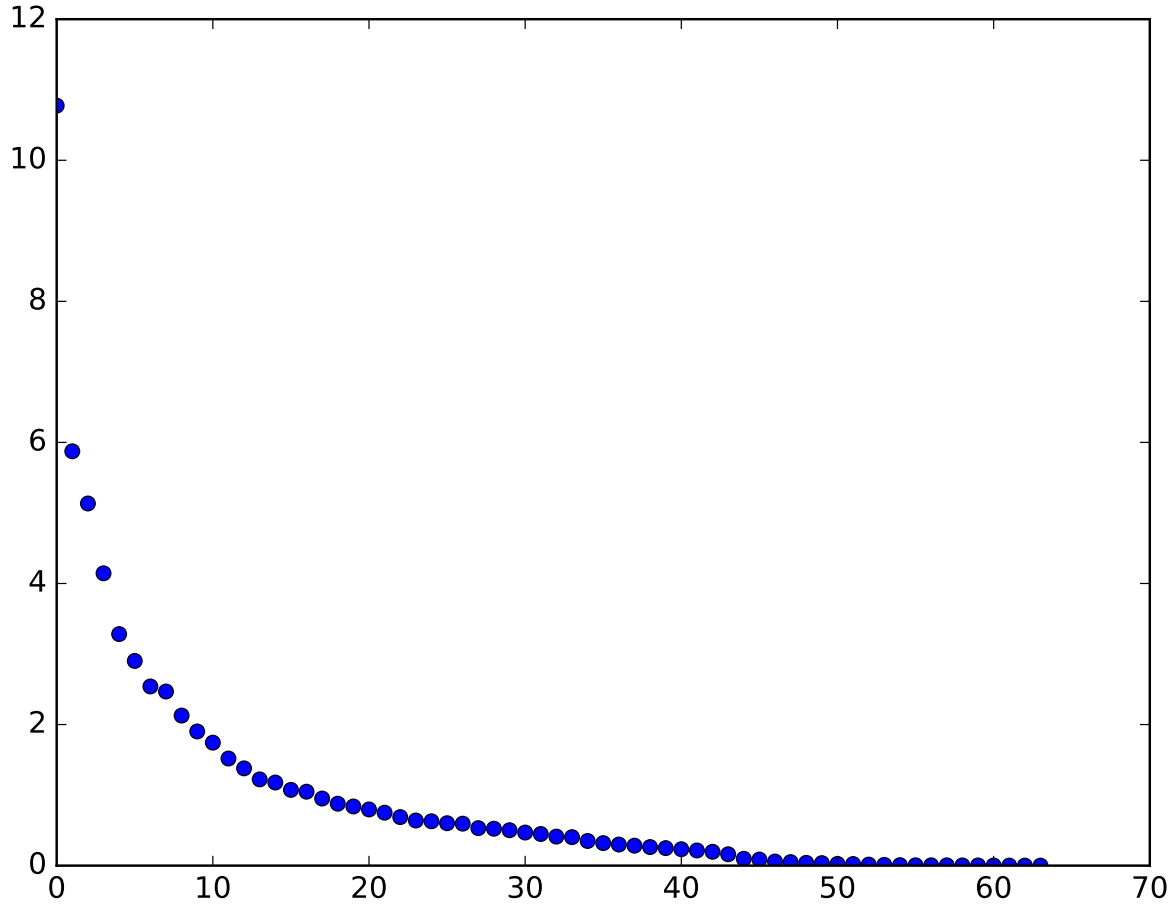
Let $S = \text{diag}(\|\mathbf{\Phi}_{\bullet 1}\|_2, \ldots, \|\mathbf{\Phi}_{\bullet D}\|_2)$. Then:

$$\begin{aligned}\min_{w \in \mathbb{R}^D} \frac{1}{n}\|\mathbf{y} - \mathbf{\Phi} w\|_2^2 + \lambda \|w\|_2^2 &= \min_{w \in \mathbb{R}^D} \frac{1}{n}\|\mathbf{y} - (\mathbf{\Phi} S^{-1})(Sw)\|_2^2 + \lambda \|w\|_2^2 \\ &= \min_{\theta \in \mathbb{R}^D} \frac{1}{n}\|\mathbf{y} - (\mathbf{\Phi} S^{-1})\theta\|_2^2 + \lambda \|S^{-1}\theta\|_2^2 \\ &= \min_{\theta \in \mathbb{R}^D} \frac{1}{n}\|\mathbf{y} - (\mathbf{\Phi} S^{-1})\theta\|_2^2 + \lambda \sum_{j=1}^{D} \frac{w_j^2}{\|\mathbf{\Phi}_{\bullet j}\|_2^2}.\end{aligned}$$

$\rightsquigarrow$ features with small norm (scale) are penalized more strongly.

Ridge Regression

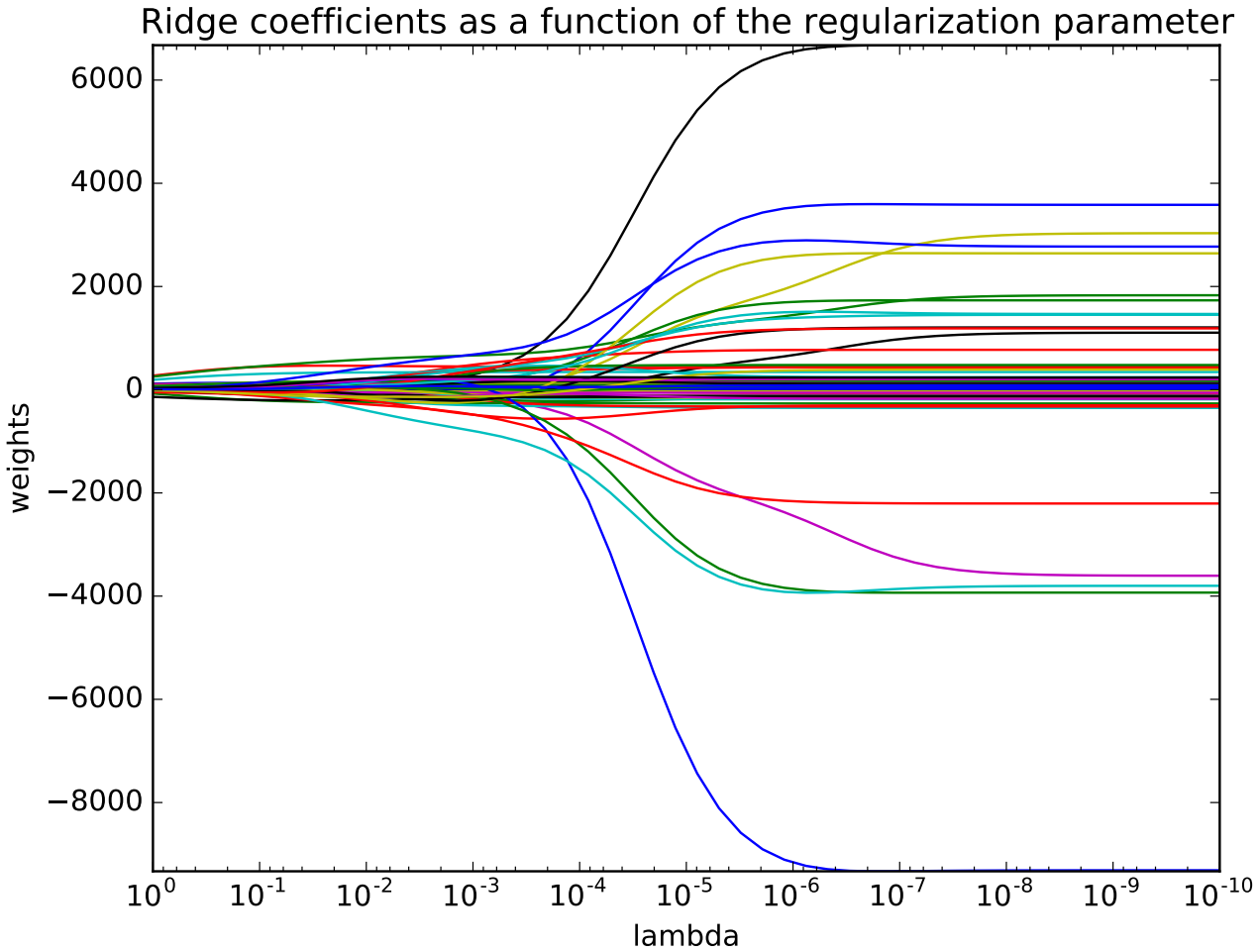
Back to the data set. "Scree plot" of the eigenvalues of $\mathbf{\Phi}^\top \mathbf{\Phi}$ (after centering / scaling):



4 prominent eigenvalues, several (~ 10) very small eigenvalues.

Ridge Regression

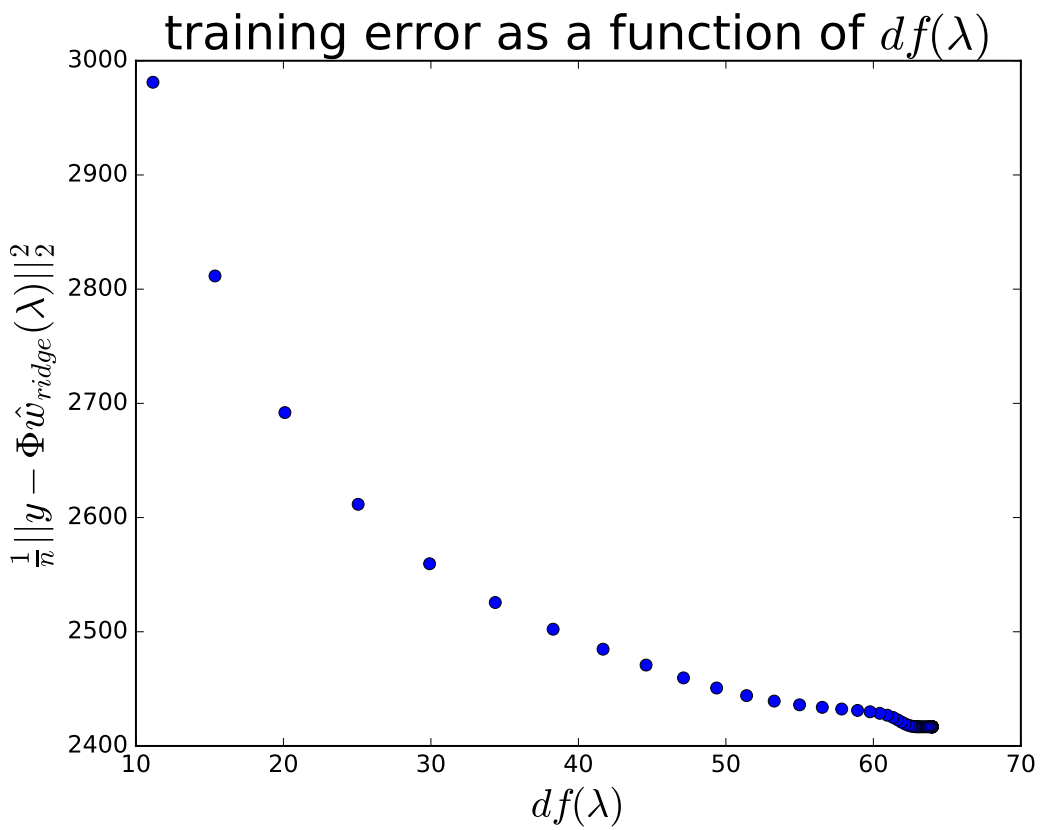
Ridge coeffients (or weights) $\widehat{w}_{\text{ridge},j}$, $j = 1, \ldots, D$, as a function of the regularization parameter:



Ridge Regression

"Effective #of variables" / "degrees of freedom" in ridge regression:

$$\text{df}(\lambda) = \sum_{j=1}^{D} \left(\frac{\lambda_j}{\lambda_j + \lambda} \right)^2$$



LASSO

Another regularization method that works better in a "sparse regime" is the lasso:

$$\widehat{w}_{\text{lasso}} \in \operatorname*{argmin}_{w \in \mathbb{R}^D} \frac{1}{2n} \|\mathbf{y} - \mathbf{\Phi} w\|_2^2 + \lambda \|w\|_1.$$

It can be motivated as a convex relaxation of

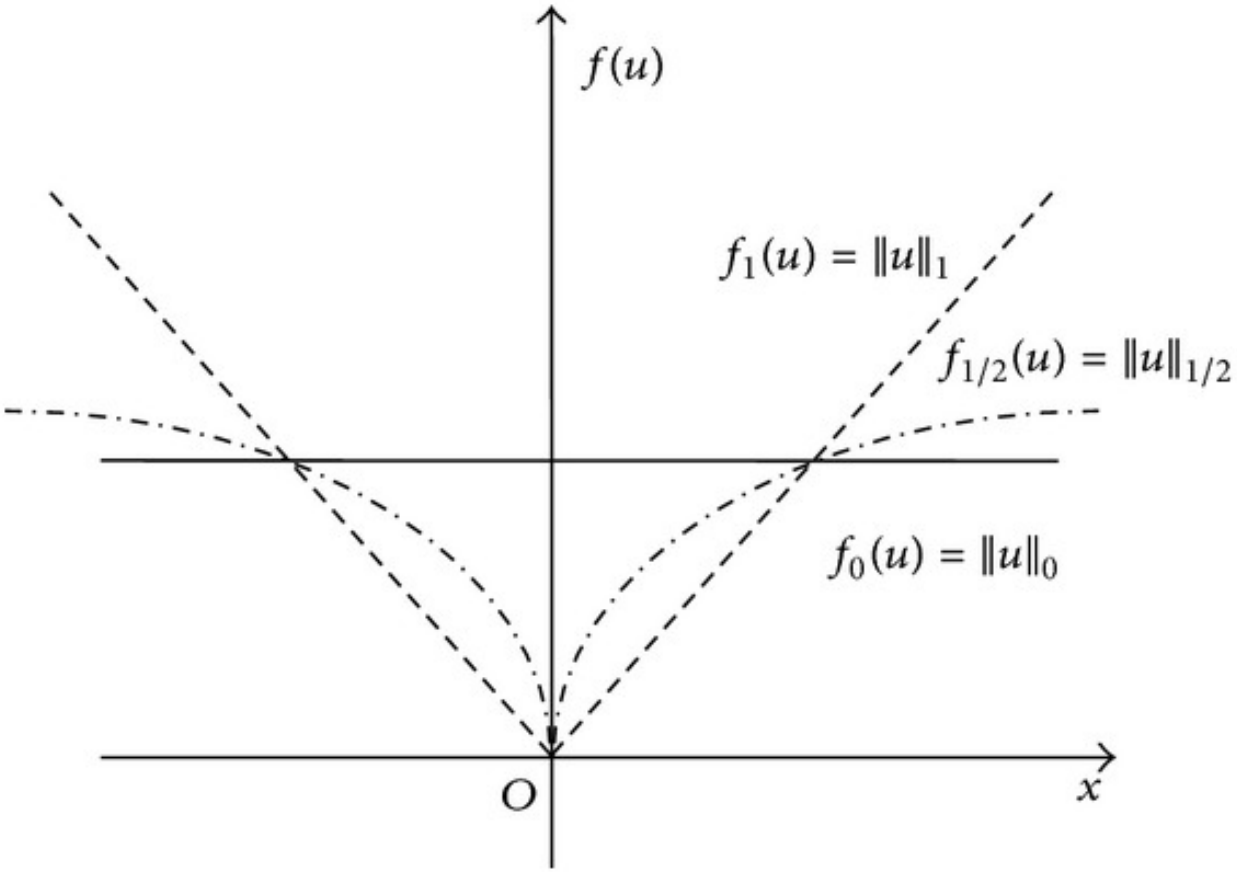
$$\widehat{w}_{\ell_0} \in \operatorname*{argmin}_{w \in \mathbb{R}^D} \frac{1}{2n} \|\mathbf{y} - \mathbf{\Phi} w\|_2^2 + \lambda \|w\|_0,$$

$$\|w\|_0 := \sum_{j=1}^{D} I(w_j \neq 0)$$

which is computationally intractable.

LASSO

$\|\cdot\|_1$ is the tightest convex approximation to $\|\cdot\|_0$ on the cube B_∞^d:



LASSO

The ℓ_1-norm regularizer produces sparse solutions, i.e., depending on λ, many entries of $\widehat{w}_{\text{lasso}}$ are exactly zero.

That is: the lasso implicitly performs feature selection.

In the case that $\frac{1}{n}\mathbf{\Phi}^\top\mathbf{\Phi} = \text{diag}(\lambda_1, \ldots, \lambda_D)$, it can be shown that

$$\widehat{w}_{\text{lasso},j} = \frac{\text{sign}\left(\frac{1}{n}\sum_{i=1}^n \Phi_j(X_i)Y_i\right) \cdot \max\{\left|\frac{1}{n}\sum_{i=1}^n \Phi_j(X_i)Y_i\right| - \lambda, 0\}}{\lambda_j},$$

$j = 1, \ldots, D$.

This is known as "Soft Thresholding" (Donoho and Johnstone, 1994).

LASSO

In the case that $\frac{1}{n}\mathbf{\Phi}^\top\mathbf{\Phi} = \text{diag}(\lambda_1, \ldots, \lambda_D)$, it can be shown that

$$\widehat{w}_{\text{lasso},j} = \frac{\text{sign}\left(\frac{1}{n}\sum_{i=1}^n \Phi_j(X_i)Y_i\right) \cdot \max\{\left|\frac{1}{n}\sum_{i=1}^n \Phi_j(X_i)Y_i\right| - \lambda, 0\}}{\lambda_j},$$

$j = 1, \ldots, D$.

In particular, this implies that

$$\left|\frac{1}{n}\sum_{i=1}^n \Phi_j(X_i)Y_i\right| < \lambda \quad \Longrightarrow \widehat{w}_{\text{lasso},j} = 0, \quad j = 1, \ldots, D.$$

LASSO

Hard thresholding (Left) vs. Soft Thresholding (Right)